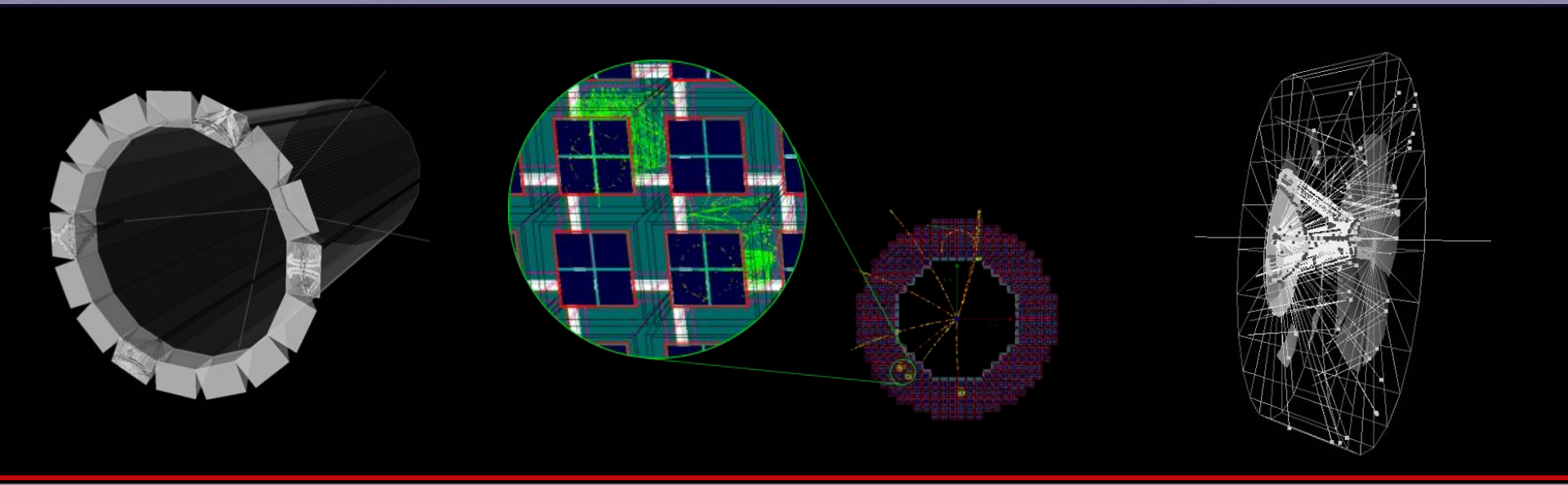
Artificial Intelligence for Imaging Cherenkov Detectors

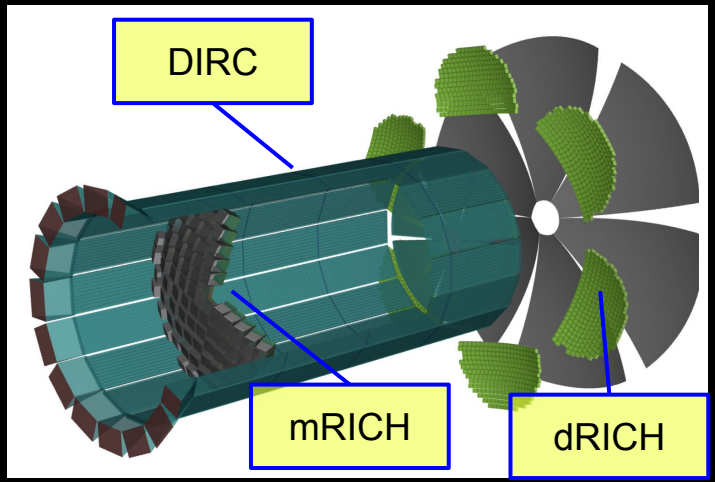


Cristiano Fanelli

EIC PID with Cherenkov

See talk by S. Joosten

η	θ	Nomenclature			Tracking: Resolution	Tracking: Relative Momentum	Tracking: Allowed X/X0	Tracking: Minimum-pT	Tracking: Transverse Pointing Res.	Tracking: Longitudinal Pointing Res.	Electrons and Photons: Resolution σE/E	Electrons and Photons: PID	Electrons and Photons: Min E Photon	π/K/p: p-Range (GeV/c)	π/K/p: Separation	HCAL: Resolution σE/E	HCAL: Energy	Muons
< -4.6		↓ p/A	Far Backward Detectors	low-Q2 tagger														
-4.6 to -4.0					Not Accessible													
-4.0 to -3.5					Reduced Performance													
-3.5 to -3.0			Backward Detector			σp/p ~0.2%⊕p⊕5%	~5% or less X	70-150 MeV/c (B=1.5 T)			1%/E ⊕ 2.5%/√E ⊕ 1%	π suppression up to 1:1E-4	20 MeV	≤ 10 GeV/c	≥ 3 σ	50%/√E⊕10%	~500MeV	Muons useful for bkg. improve resolution
-3.0 to -2.5																		
-2.5 to -2.0						σp/p~ 0.04%⊕p⊕2%			dca(xy) ~ 40/pT μm ⊕ 10 μm	dca(z) ~ 100/pT μm ⊕ 20 μm	2%/E ⊕(4-8)%/√E ⊕ 2%	π suppression up to 1:1E-3 - 1E-2)	50 MeV					
-2.0 to -1.5																		
-1.5 to -1.0																		
-1.0 to -0.5			Barrel			σp/p ~0.04%⊕p⊕1%		200 MeV/c	dca(xy) ~ 30/pT μm ⊕ 5 μm	dca(z) ~ 30/pT μm ⊕ 5 μm	2%/E ⊕(12-14)%/√E ⊕ (2-3)%	π suppression up to 1:1E-2	100 MeV	≤ 6 GeV/c		100%/√E+10%		
-0.5 to 0.0																		
0.0 to 0.5																		
0.5 to 1.0																		
1.0 to 1.5			Forward Detectors			σp/p ~0.04%⊕p⊕2%		70 - 150 MeV/c (B = 1.5 T)	dca(xy) ~ 40/pT μm ⊕ 10 μm	dca(z) ~ 100/pT μm ⊕ 20 μm	2%/E ⊕ (4*-12)%/√E ⊕ 2%	3σ e/π up to 15 GeV/c	50 MeV	≤ 50 GeV/c		50%/√E+10%		
1.5 to 2.0																		
2.0 to 2.5																		
2.5 to 3.0																		
3.0 to 3.5																		
3.5 to 4.0				Instrumentation to separate charged particles from photons		Reduced Performance												
4.0 to 4.5		↑ e			Not Accessible													
> 4.6			Far Forward Detectors	Proton Spectrometer														
				Zero Degree Neutral Detection														



- Cherenkov detectors form the backbone of PID at EIC
- Currently, all EIC detector designs use a dual radiator ring-imaging Cherenkov detector (RICH) in the hadron direction, a DIRC (detection of internally reflected Cherenkov light) in the barrel, and a modular RICH in the electron direction.
- In his talk Sylvester highlighted how simulating these detectors is typically compute expensive, involving many photons that need to be tracked through complex surfaces.
- All three rely on pattern recognition of ring images in reconstruction, and the DIRC is the one having the more complex ring patterns! In the following I will use the DIRC as an example.

Image of expansion volume taken from GlueX DIRC, Ali et al., JINST 15 (2020) 04, C04054

Example from GlueX

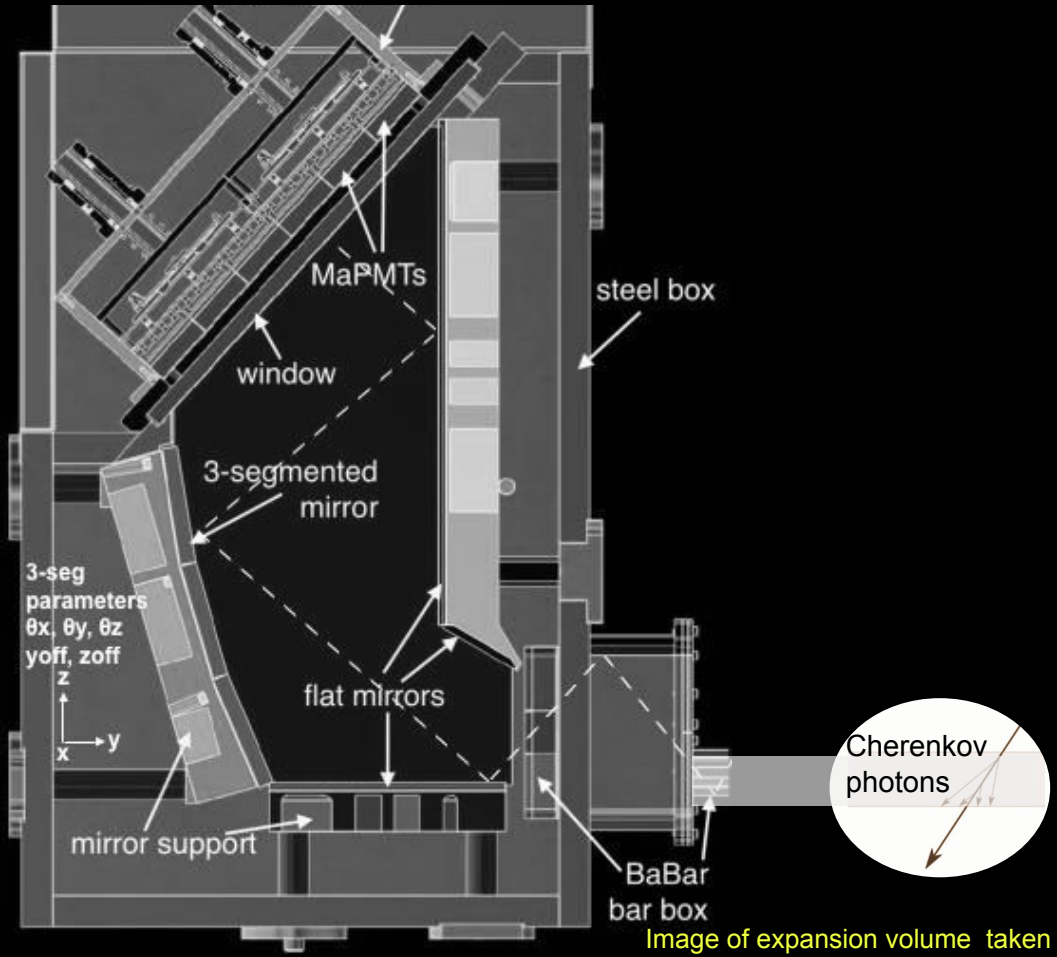
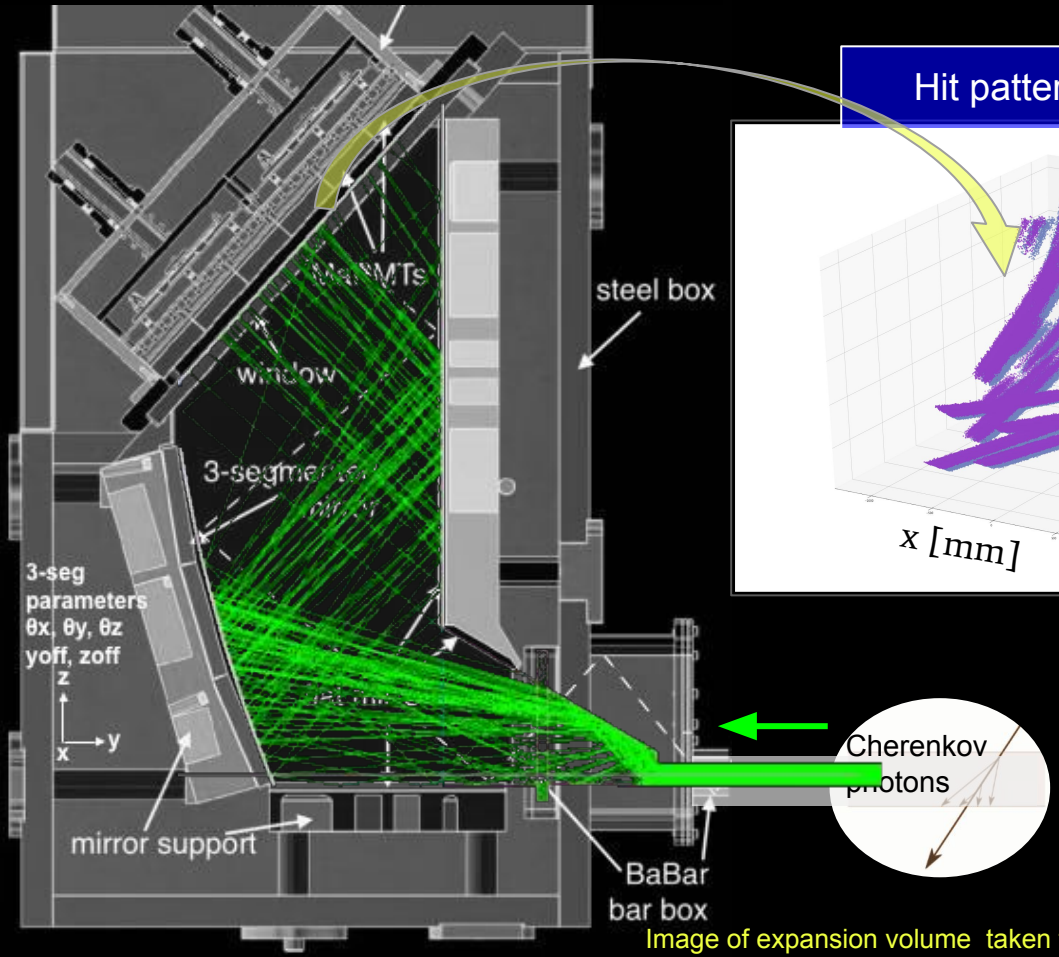
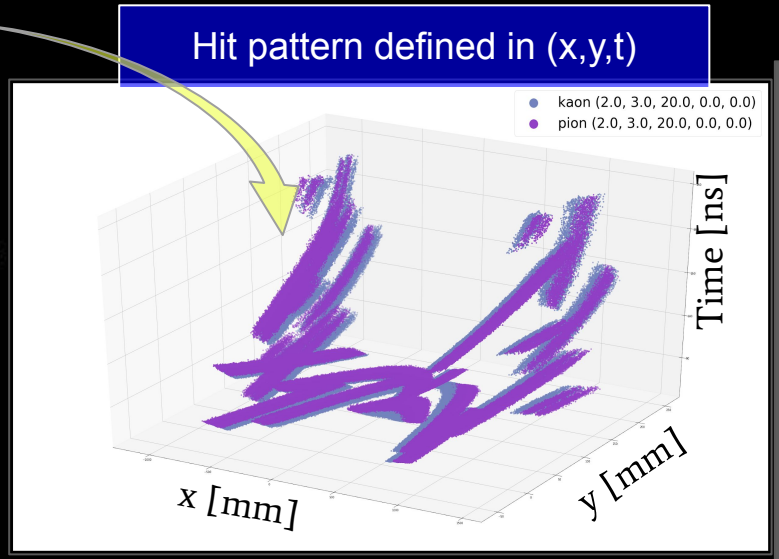


Image of expansion volume taken from GlueX DIRC, Ali et al., JINST 15 (2020) 04, C04054

Image of expansion volume taken from GlueX DIRC, Ali et al., JINST 15 (2020) 04, C04054

Example from GlueX





3D (x,y,t) readout allows to separate spatial overlaps.

Patterns take up significant fractions of the PMT in x,y and are read out over 50-100 ns due to propagation time in bars.

H12700 PMTs have a time resolution of O(200 ps) and read-out electronics giving time information in 1 ns buckets.

1PMT made by 64 pixels, each pixel is 6mm x 6mm size

Displayed PDF. Patterns are sparse with variable photon yield

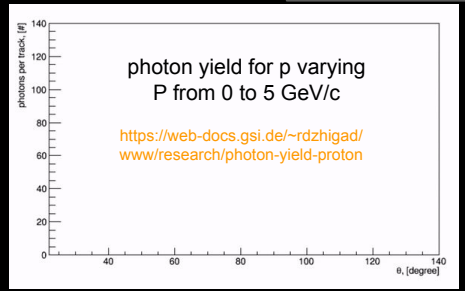
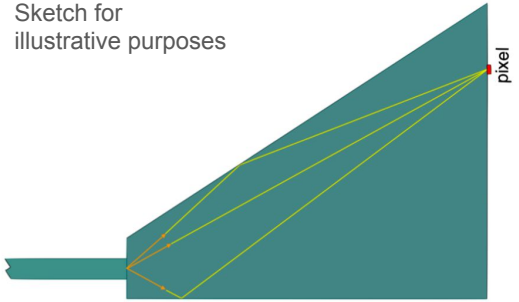
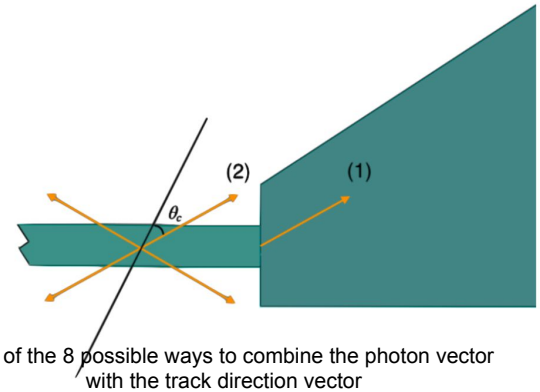


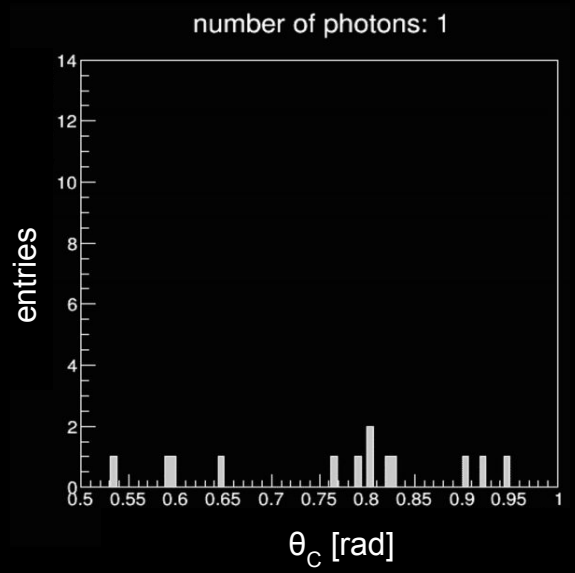
Image of expansion volume taken from GlueX DIRC, Ali et al., JINST 15 (2020) 04, C04054

Geometrical Reconstruction

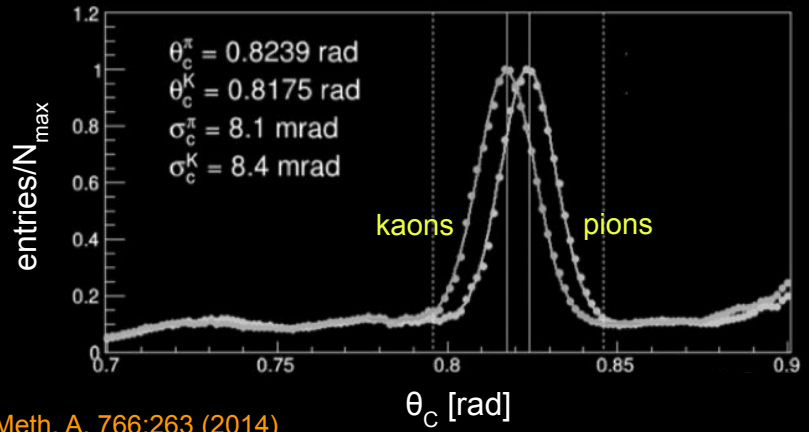




Four of the 8 possible ways to combine the photon vector with the track direction vector



- All possible photon paths from the bar to each pixel are stored in look-up tables
- The Cherenkov angle is determined by calculating the angle between the photon direction from the LUT and the charged track direction from the tracking system
- Fast reconstruction/hit pattern
- Other approaches possible (e.g., time-based imaging utilizes detection time per pixel; superior reconstruction but memory typically hungry)



R. Dzhygadlo et al. Nucl. Instr. And Meth. A, 766:263 (2014)

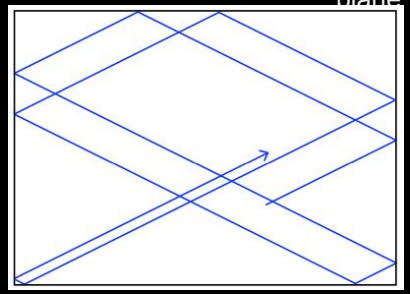
FastDIRC

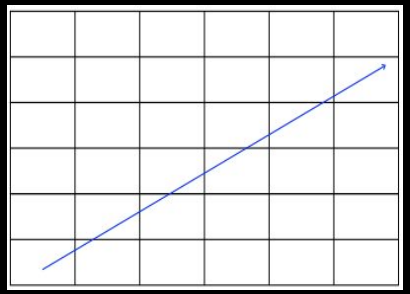
J. Hardin and M. Williams, JINST 11.10 (2016)

https://github.com/jmhardin/FastDIRC

- Framework for Fast Monte Carlo and reconstruction.
- Simulations: fast tracing mapping straight lines through a tiled plane:
  1. Generation
  2. Traces through bars
  3. Traces through expansion volume

Each photon bounces O(100) times on average. Developed a billiard method that maps the bounces onto a straight-line trajectory through a tiled plane.





10000 faster than Geant4:
This facilitates reconstruction of Cherenkov angle with an improvement of 30% as compared to the Geometrical approach.
Slower than LUT.

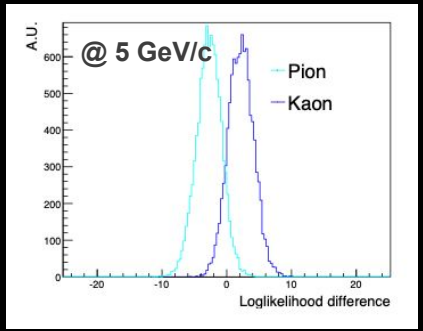
- PID strategy is likelihood based:
  - N_g photons are generated to produce the expected PDF.
  - N_g and λ chosen to provide best performance.

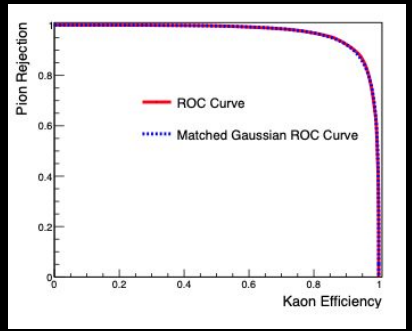
$$f(\vec{x}) \propto \sum_{i}^{n_\gamma^{\mathrm{sim}}} \mathscr{G}(|\vec{x}-\vec{x}_i|)$$

PDF

$$\log \mathcal{L}_{\pi(K)} = \sum_{j=1}^{N_d} \ln\Big(\sum_{i=1}^{N_g^{\pi(K)}} g\Big(\frac{|\mathbf{x}_i^{\pi(K)} - \mathbf{x}_j|}{\lambda}\Big)\Big)$$

likelihood



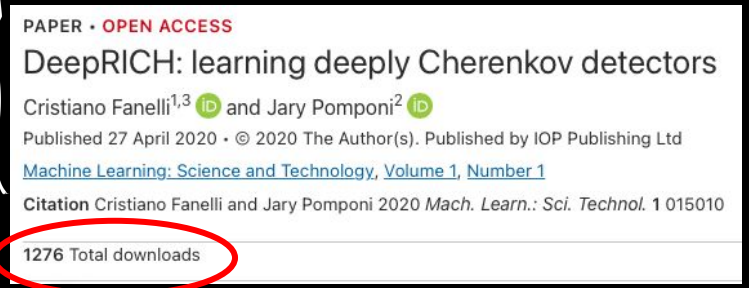


Machine Learning for Cherenkov Detectors?

- Cherenkov detectors are relatively slow to simulate with full simulations like Geant
  - for the DIRC case, each Cherenkov photon reflects on average O(10^2) times within a bar and this makes the simulation CPU intensive.
- Not many AI-based applications:
  - Some work on fast simulation with Cherenkov detectors [1].
  - Lack of ML/DL applications for reconstruction/identification:
    - Most of them use high-level features from Cherenkov detectors and combine them to other features from other sub-detectors for global PID [2].
- Can we build an AI-based architecture with the following desired properties?

- It is fast* and provides accurate reconstruction
- Can be extended to multiple particle types
- Generalizes to fast simulation
- Can utilize (x,y,t) patterns if time is measured
- Can deal with different topologies and detectors
- Deeply learns the detector response (real data can be injected)

DeepRICH [3] is the first attempt in this direction.
We'll show prototype and discuss path forward.



[1] D. Derkach et al., arXiv:1903.11788v1, 2019
[2] D. Derkach et al., J. Phys.: Conf. Ser. **1085** 042038, 2018
[3] C. Fanelli and J. Pomponi, Mach. Learn.: Sci. Technol. 1 015010, 2020

DeepRICH Architecture

CF and J. Pomponi, Mach. Learn.: Sci. Technol. 1 015010, 2020

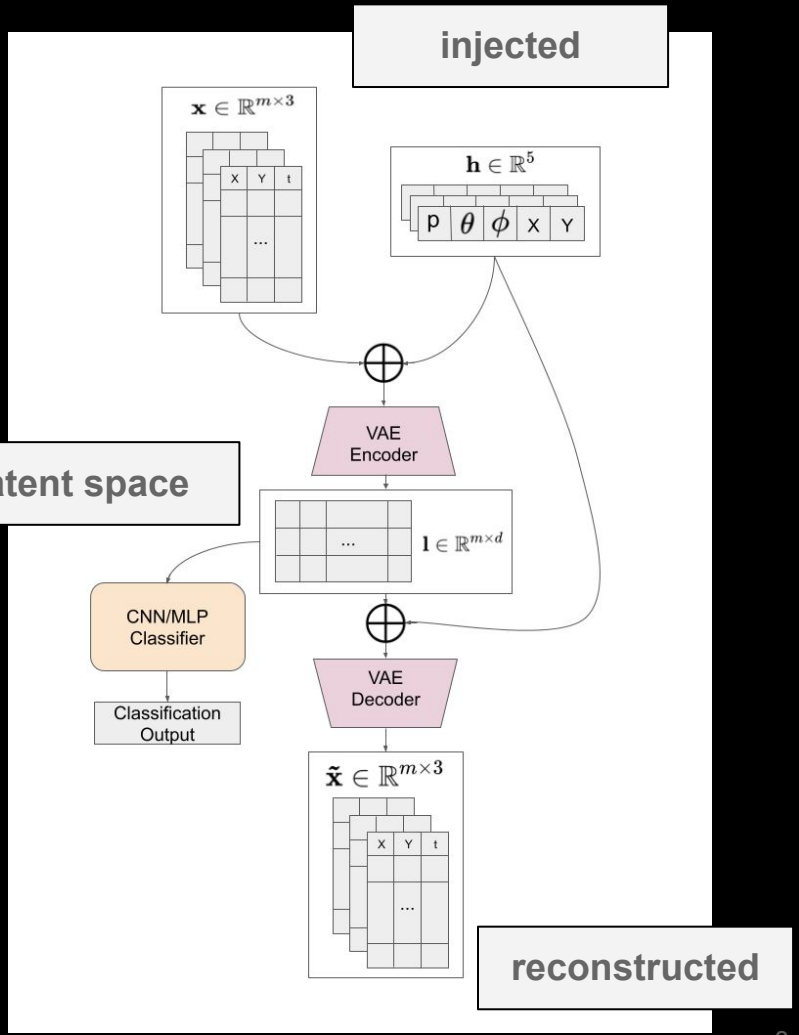
- DeepRICH is a custom architecture that combines
  - VAE for reconstruction
  - CNN + MLP for classification
- The model is trained by minimizing the total loss function:

$$\mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}}, \mathbf{y}, \tilde{\mathbf{y}}, \mathbf{l}) = \lambda_r \mathcal{L}_r(\mathbf{x}, \tilde{\mathbf{x}}) + \lambda_c \mathcal{L}_c(\mathbf{y}, \tilde{\mathbf{y}}) + \lambda_v \mathcal{L}_v(\mathbf{l})$$

Reconstruction loss (injected vs reco)

Classification loss

VAE MMD (latent)



DeepRICH Architecture

CF and J. Pomponi, Mach. Learn.: Sci. Technol. 1 015010, 2020

- DeepRICH is a custom architecture that combines
  - VAE for reconstruction
  - CNN + MLP for classification
- The model is trained by minimizing the total loss function:

$$\mathcal{L}(\mathbf{x},\tilde{\mathbf{x}},\mathbf{y},\tilde{\mathbf{y}},\mathbf{l}) = \lambda_r\mathcal{L}_r(\mathbf{x},\tilde{\mathbf{x}}) + \lambda_c\mathcal{L}_c(\mathbf{y},\tilde{\mathbf{y}}) + \lambda_v\mathcal{L}_v(\mathbf{l})$$

Reconstruction loss (injected vs reco)

$$\mathcal{L}_r(x,\tilde{x}) = \frac{1}{3}\sum_i^3 z_i$$

$$z_i = \begin{cases} 0.5(x_i-\tilde{x}_i)^2, & \text{if } |x_i-\tilde{x}_i|<1 \\ |x_i-\tilde{x}_i|-0.5 & \text{otherwise,} \end{cases}$$

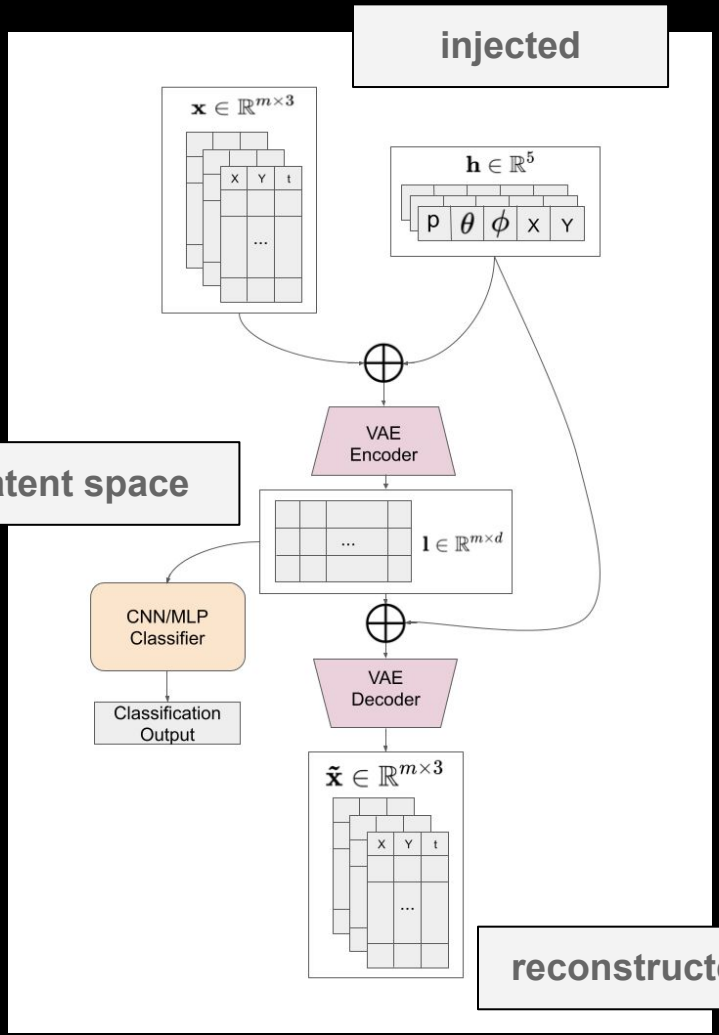
Classification loss

$$\mathcal{L}_c = -(\mathbf{y}\log(\tilde{\mathbf{y}}_0) + (1-\mathbf{y})\log(\tilde{\mathbf{y}}_1))$$

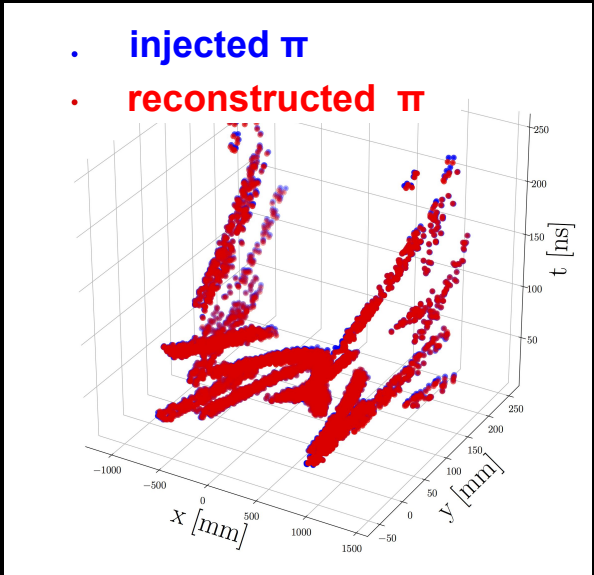
tackled π, K separation. This can be extended to, e.g., p

VAE MMD (latent)

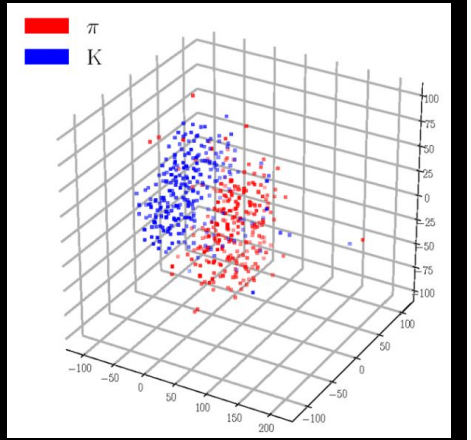
$$\mathcal{L}_v = MMD(\boldsymbol{p}(z),\boldsymbol{q}(z)) = \mathbb{E}_{\boldsymbol{p}(z),\boldsymbol{p}(z')}[\kappa(z,z')] + \mathbb{E}_{\boldsymbol{q}(z),\boldsymbol{q}(z')}[\kappa(z,z')] - 2\mathbb{E}_{\boldsymbol{p}(z),\boldsymbol{q}(z')}[\kappa(z,z')]$$



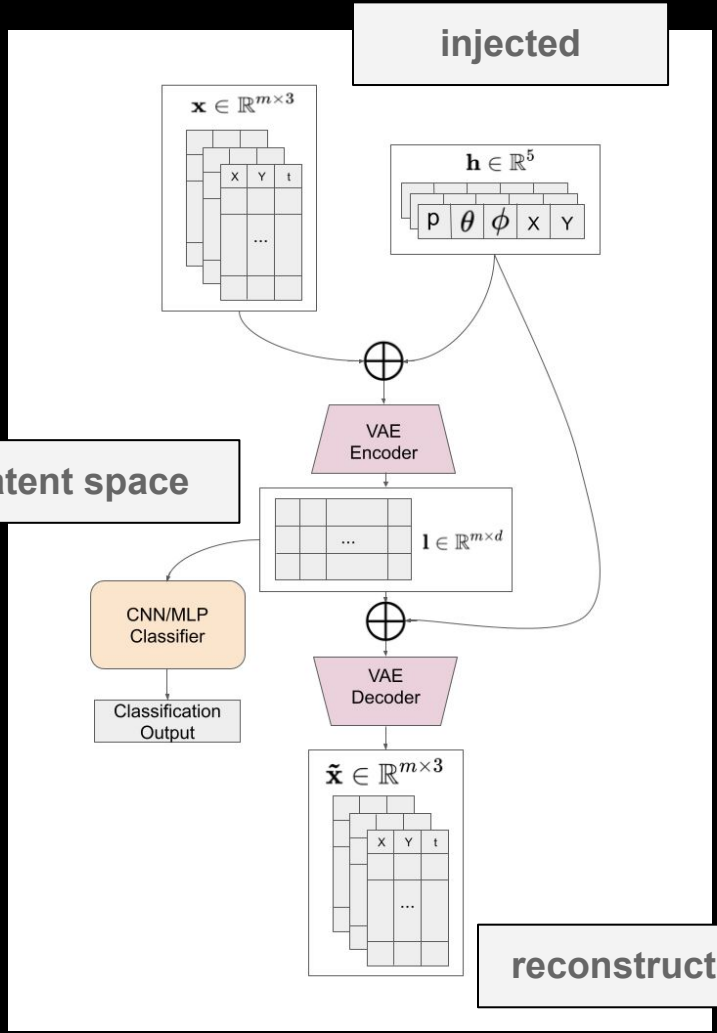
DeepRICH Architecture



Example of hit pattern detected in the PMT plane (spatial coordinates are dubbed x,y, while the time is indicated as t) simulated with FastDIRC and reconstructed with VAE



Example of features extracted by the CNN from pi and K at 5 GeV. The plot shows separation power. The 3D visualization is obtained with t-SNE.



Data Preparation

1. Classification is supervised and needs labeled data:
   - Real data: High purity samples can be created using specific topologies with π, K.
   - Simulated: Geant simulations (compute expensive) or fast and accurate simulations (FastDIRC).

2. Been tested on data generated with FastDIRC
   - (an established approach for (1) simulation + (2) reconstruction), which is orders of magnitude faster than Geant in simulating the hit patterns observed in the PMT detection plane.

3. Built a PDF with FastDIRC
   - At a given kinematics of the track (P,θ,φ,X,Y) FastDIRC generates ~10^5 "provisional" points (x,y) in the PMT plane from which one obtains a PDF. FastDIRC can also simulate the "real" detected particles which produce sparse hit patterns. We studied photon yields of m~20-50 hits. (dataset)

4. Generator-independent (and virtually unlimited) dataset:
   - At each (P,θ,φ,X,Y) we use the "PDF" (one for π, one for K), and then sample N particles of a given type made by a random set of m hits. This allows to build a virtually unlimited dataset and also avoid learning any patterns internal to the FastDIRC generation algorithm. ("unlimited" dataset)

5. Closure Test
   - We compare the reconstruction performance of DeepRICH vs FastDIRC on our dataset. Check consistency of the FastDIRC performance (vs kinematics) using independent datasets.

Training and Testing

- Prototyped with a large dataset corresponding to ~$2 \cdot 10^4$ kinematic points created to cover the subspace
  $\Delta p \times \Delta\theta \times \Delta\varphi \times \Delta X \times \Delta Y$ = [4,5] GeV/c × [2,5] deg × [20,90] deg × [-17.5,17.5] mm x [50, 1000] mm
  where p, θ, φ, X, Y have been divided into equally distant points within those intervals.
- A more dense grid of points combined with a larger number of sampled particles at each kinematic point generally improves the PID performance.
- The generated samples have been then divided into two subsets:

  - Training Set: contains particles at certain kinematics used during the training phase --- we include the "vertices" of the hypercube in the training. The dataset is divided in:
    - 'Training particles' (80%) update the network parameters by minimizing the total loss
    - 'Development particles' (20%) calculate an accuracy score and check if the network is learning properly. Early stopping is used to interrupt the training if the development score does not improve after a certain number of epochs.
  - Test Set: used to test if the network provides good performance on unknown kinematics

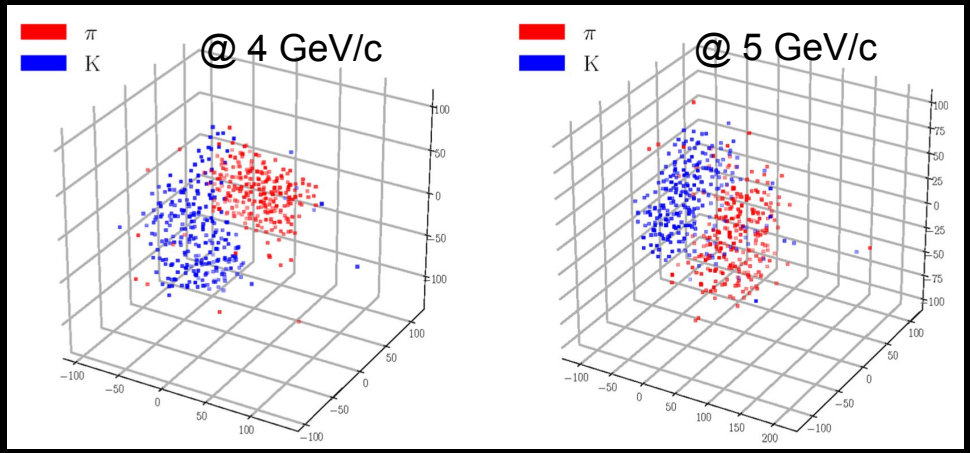
- We used Bayesian Optimization to tune the multipliers of the loss function, the dimension of the latent space, the MMD variance and the learning rate. Each call based on 50 epochs.

Table 2. List of hyperparameters tuned by the BO. The tuned values are shown in the outermost right column. The optimized test score is about 92%.

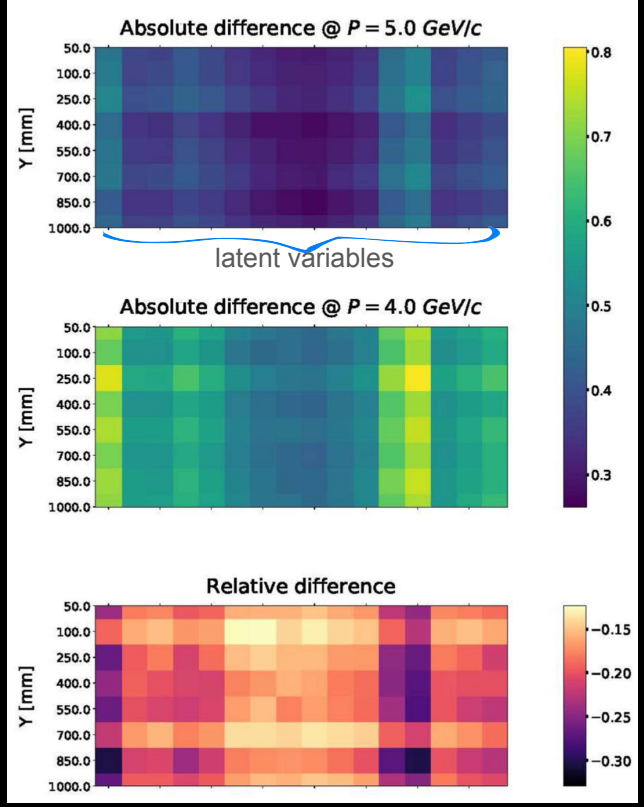
symbol	description	range	optimal value
`NLL`	λ_r	$[10^{-1},10^{2}]$	0.784
`CE`	λ_c	$[10^{-1},10]$	1.403
`MMD`	λ_v	$[1,10^{3}]$	1.009
`LATENT_DIM`	latent variables dimension	[10,200]	16
`var_MMD`	σ in $\mathcal{N}(0,\sigma)$	[0.01,2]	0.646
`Learning Rate`	learning rate	[0.000 1, 1]	$6.6 \cdot 10^{-4}$

Results

π/K distinguishing power: visualizations



Example of features extracted by the CNN module from π's and K's at 4 GeV/c (left) and 5 GeV/c (right). These features are then used to classify the particle. The plot shows a better separation between π/*K* at 4 GeV/c, which means that the network has good distinguishing power. As expected the points become less separated at larger momentum. The 3D visualization is obtained with t-SNE.



2D plot of the absolute difference on each latent variable between π's and K's, obtained for 5 GeV/c and 4 GeV/c, respectively. The color indicates the absolute difference, the larger the difference the larger is the distinguishing power. As expected the separation becomes less clear at 5 GeV. Also there is no appreciable dependence on the position on the bar resulting in patterns with vertical bands. (Bottom) The relative difference showing negative values in the majority of the bins.

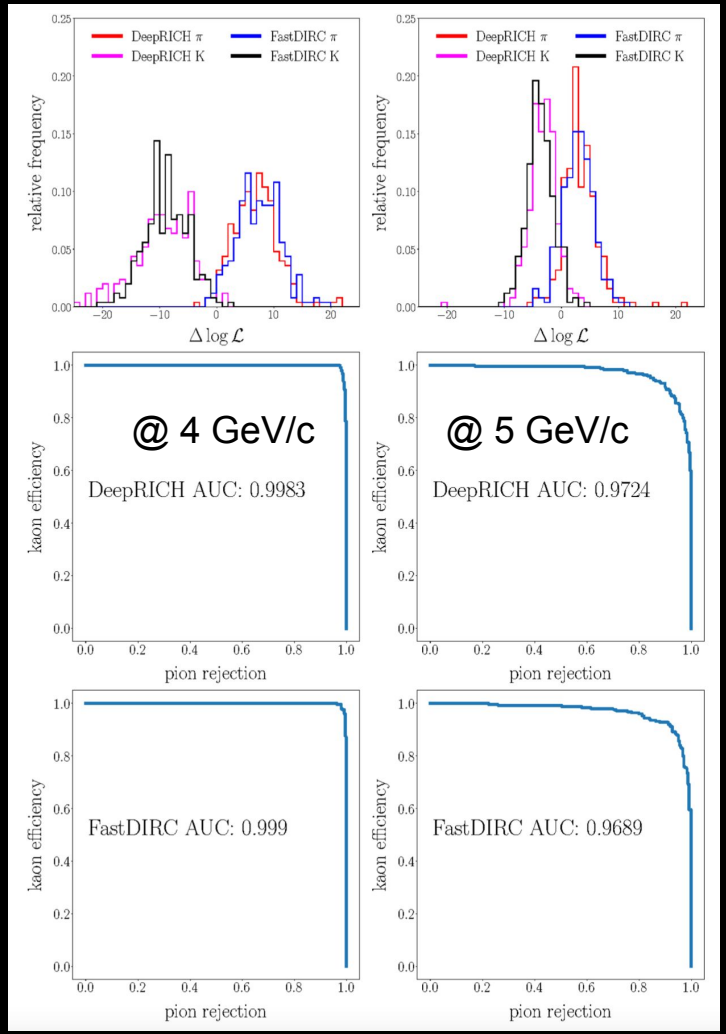
Results

$$\log \mathcal{L}_{\pi(K)} = \sum_{j=1}^{N_d} \ln \left(\sum_{i=1}^{N_g^{\pi(K)}} g\left(\frac{|\mathbf{x}_i^{\pi(K)} - \mathbf{x}_j|}{\lambda} \right) \right)$$

In DeepRICH the output of the classifier is two-dimensional (π/K) and $\in R^2$.

- These values are utilized to build the DLL between π/K
- ROC is obtained by changing the threshold on the DLL. ROC curves are produced generating 350 particles for each kinematics
- The AUC is used as a metric to compare DeepRICH to FastDIRC

AUC(DeepRICH) ≳ 0.99 AUC(FastDIRC)

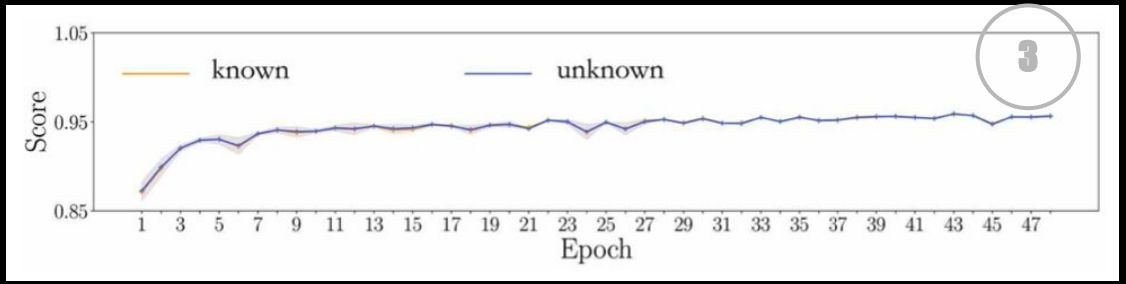


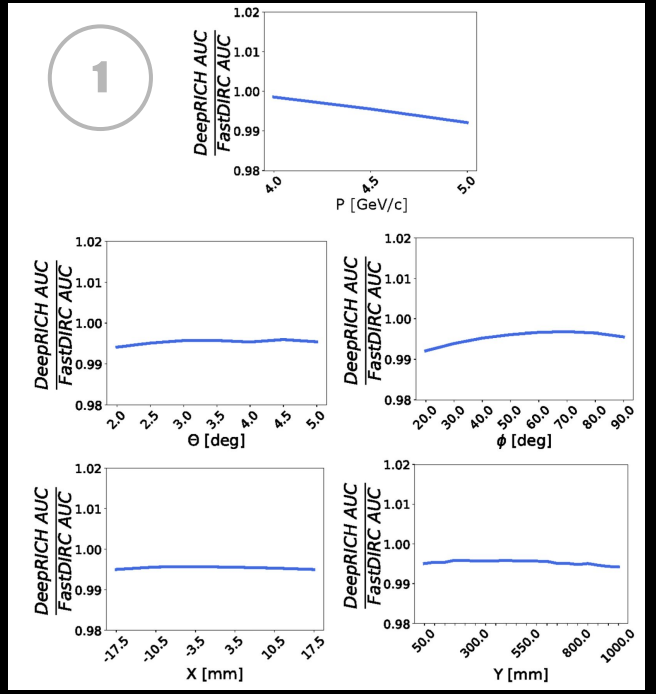
Results

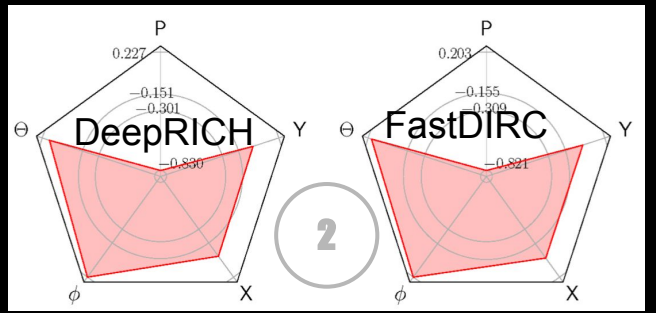
(1) The ratio between DeepRICH and FastDIRC AUCs: Each AUC is calculated to show the partial dependence on one kinematic parameter by marginalizing on all other parameters. Notice at 4 GeV/c that the two reconstruction methods perform almost identically.

(2) Radar plots of correlation between the AUC and each kinematics parameter for DeepRICH and FastDIRC. The two reconstruction algorithms perform similarly as a function of the kinematic parameters. AUC depends on momentum, as distinguishing power gets lower at larger momentum.

(3) Learning Curve: learning curve corresponding to known and unknown kinematics combining the datasets with momentum ∈ [4,5] GeV/c. Each point is obtained as an average over 3 experiments—notice in some experiment the early stopping activated earlier. These results prove the ability of DeepRICH to reconstruct unknown kinematics.



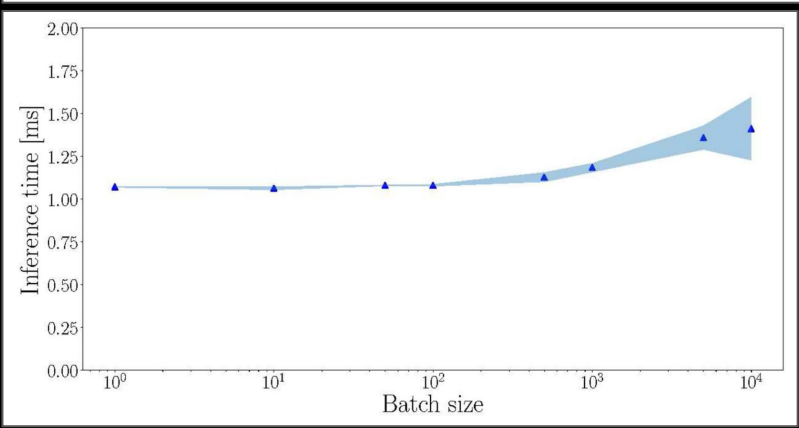




Performance

Kinematics	DeepRICH AUC	DeepRICH ε_S	DeepRICH ε_B	FastDIRC AUC	FastDIRC ε_S	FastDIRC ε_B
4 GeV/c	99.74	98.18	98.16	99.88	98.98	98.85
4.5 GeV/c	98.78	95.21	95.21	99.22	96.33	96.32
5 GeV/c	96.64	91.13	91.23	97.41	92.40	92.47

specs	value
inference time per batch	$\mathcal{O}(1)$ ms
inference network memory	$\mathcal{O}(1)$ GB
training network memory	$\mathcal{O}(4)$ GB
network memory on local storage	$\sim$ 6 MB
network trainable parameters	458 592



After training, the inference time is almost constant as a function of the batch size, meaning that the effective inference time—*i.e.*, the reconstruction time per particle—can be lower than a µs, the architecture being able to handle 10^4 particles in about 1.4 *ms* in the inference phase. Notice that the corresponding memory size in the inference phase is approximately equal to the value reported in the table.

Conclusions

- DeepRICH developed for π/K can be extended to multiple classes (e.g., p, K, π)...
- Can also be extended to larger phase space and to include entire bar box.
  - Remind that the size of the network is related to the weights and the dimensions of the architecture.
  - The training time of DeepRICH has not been optimized. One can improve this in different ways, e.g., sparser hypercube, distributed training. Without optimization, this time can be as large as 1/2 day with the described configuration on a single Titan V GPU.
- Start dedicated study as generative model for Fast Simulation.
- Performance will be characterized and benchmarked against other methods using different datasets (other than FastDIRC); in perspective one can train on high purity samples from real data.
- Topology independent. Can be adapted for other imaging detectors.

- Cherenkov detectors are the backbone of PID at EIC.
- DeepRICH is one of the few DL-approaches in the market (if not the only one) dedicated to PID with imaging Cherenkov detectors that showed promising results with a custom architecture.
- Prototype developed and proof of principle shown using simulated data in a limited phase space. Lot of exciting work is planned for the near future to further extend/improve it, see above.

Spares